CRAY SSD-T90: Interconnect Network Operations

Record of Revision

February 1997

Original printing.

Introduction

When an interconnect network failure occurs, you may need to replace a field replaceable unit (FRU) or map out a bad interconnect network component. An FRU for an interconnect network failure may be a processing element module or a ribbon cable.

This document describes the components of the interconnect network and how the components work together to transfer a packet from one PE to another PE. This document also lists the diagnostic tests you will run while troubleshooting the interconnect network. Specifically, this document answers the following questions:

1. How do the nodes communicate with each other?
2. What types of information routing does the CRAY SSD-T90 device use to transfer packets through the interconnect network?
3. What hardware components make up the interconnect network?
4. What are the three numbers that software assigns to a node?

Terms

You will need to know the following terms to fully understand the material discussed in this document:

I/O controller - The I/O controller transfers system data and control information between the CRAY SSD-T90 device and the GigaRing channel.

Node - A node contains a PE and a network router.

Processing element (PE) - A PE contains a microprocessor, local memory, and support circuitry.

Microprocessor - The microprocessor is a reduced instruction set computer (RISC) 64-bit microprocessor developed by Digital Equipment Corporation.

Support circuitry - The support circuitry is a component of a PE that extends the control and addressing functions of the microprocessor in the PE.

E registers - E registers are latency-hiding registers in the support circuitry that are the source and destination for all global data transfers.

Local memory - With respect to the microprocessor in a PE, local memory is memory that is physically located in the same PE as the microprocessor.

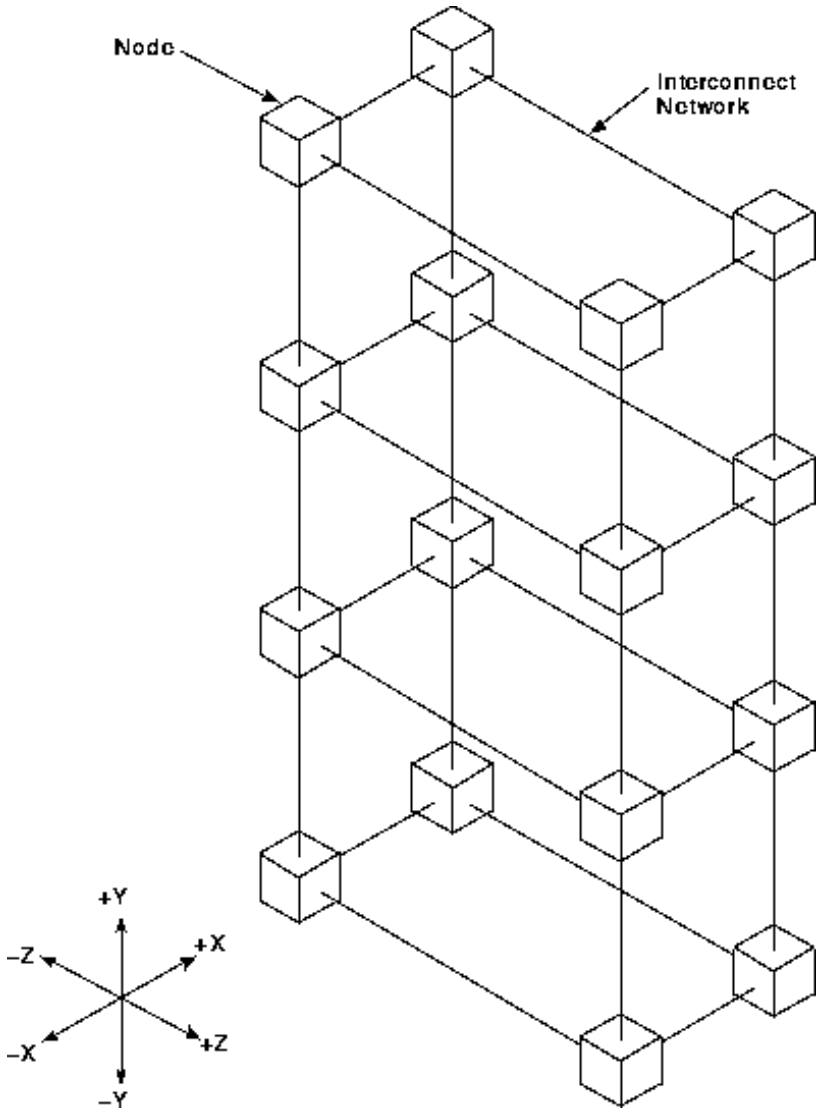
Packet - All request and response information transfers through the network in the form of a packet. A packet contains a header and a body.

Components of the Interconnect Network

Figure 1 shows a simplified model of two of the components that make up the CRAY SSD-T90 device: nodes and the interconnect network.

The interconnect network provides communication paths among the nodes in the CRAY SSD-T90 device. The interconnect network forms a three-dimensional matrix of paths that connect the nodes in the X, Y, and Z dimensions.

Figure 1. Components of the Interconnect Network

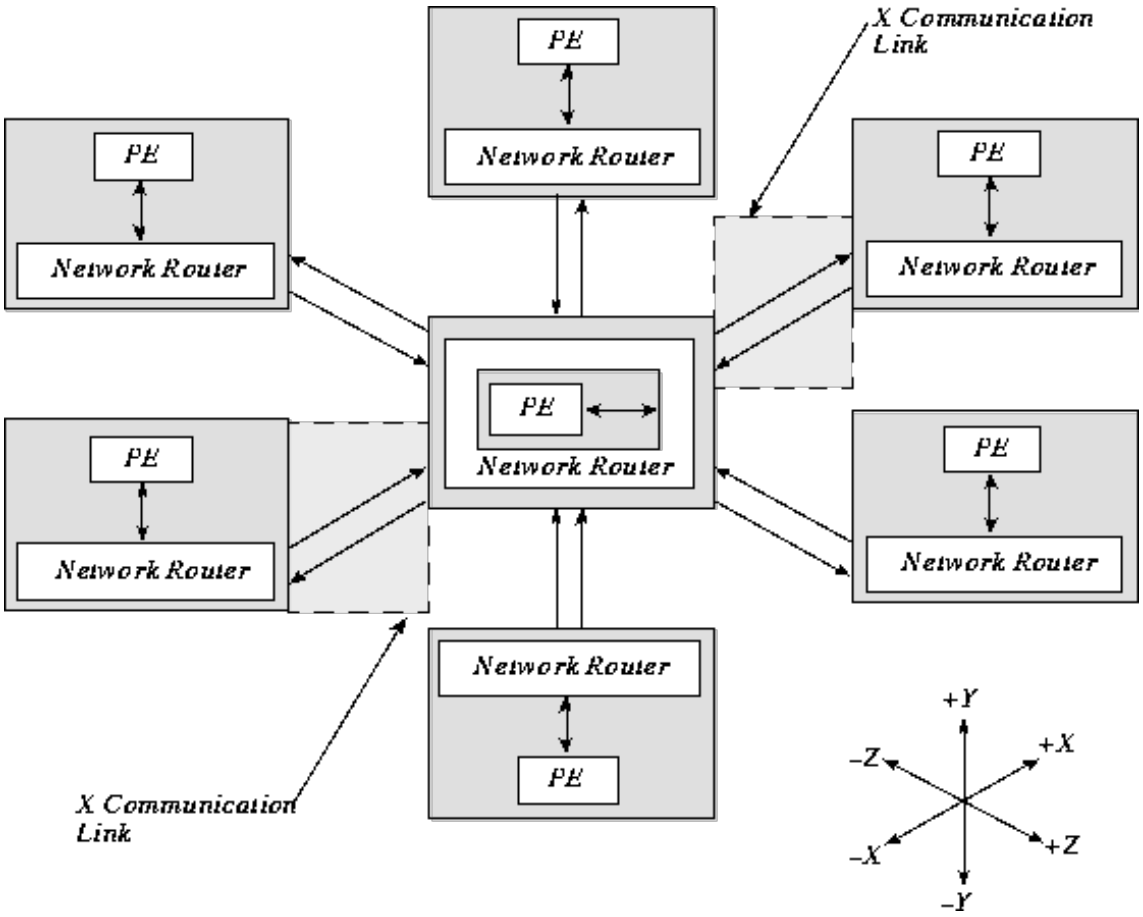


The interconnect network is composed of communication links and network routers (refer to Figure 2). The communication links transfer data and control information between the network routers. The network routers steer data and control information through the communication links.

Network Router

Each network router has connections to six communication links: +X, +Y, +Z, -X, -Y, and -Z. Each communication link consists of two unidirectional channels.

Figure 2. Components of the Interconnect Network

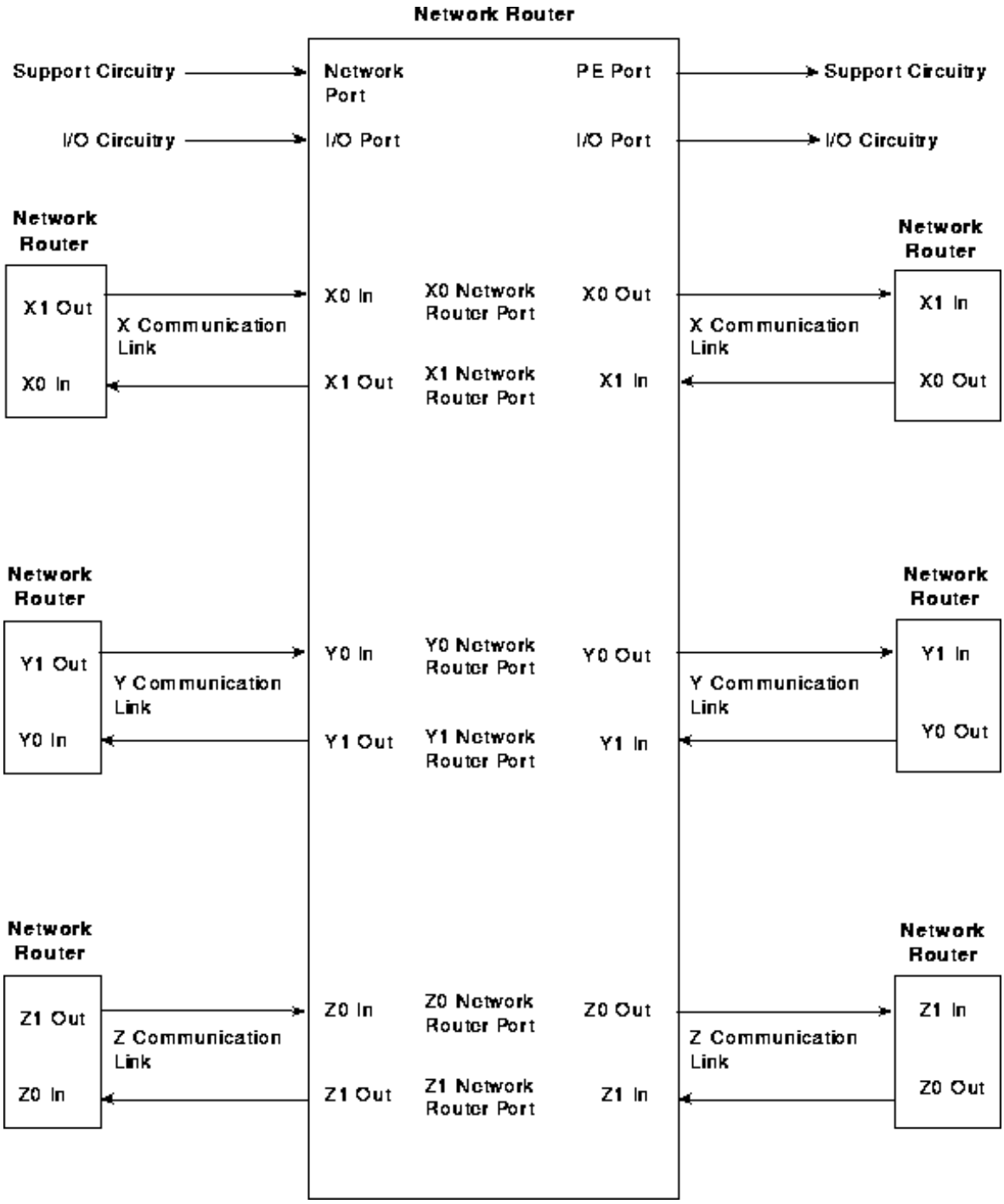


Communication Links

Communication links are connections between two network routers (refer to Figure 3). The network routers connect via network router ports. Each network router has six network router ports: X0 port, X1 port, Y0 port, Y1 port, Z0 port, and Z1 port. Each network router port has a path into and out of the network router (for example, X0 In port and X0 Out port).

NOTE: Software must assign a positive or negative orientation to each network router port. For example, software may assign the +X orientation to the X0 port and the -X orientation to the X1 port.

Figure 3. Communication Links and Network Routers



A communication link is a pair of unidirectional channels that connect between the +Out port and the -In port of one network router and the +In port and the -Out port of another network router, respectively. For example, in Figure 4 one of the X communication links is the connection between the two unidirectional channels of network router A and network router B. The other X communication link is the connection between the two unidirectional channels of network router A and network router C.

NOTE: Software writes a value to bits <2 : 0> of the R_ORIENT register to assign positive and negative orientations to the network router ports (refer to Table 1).

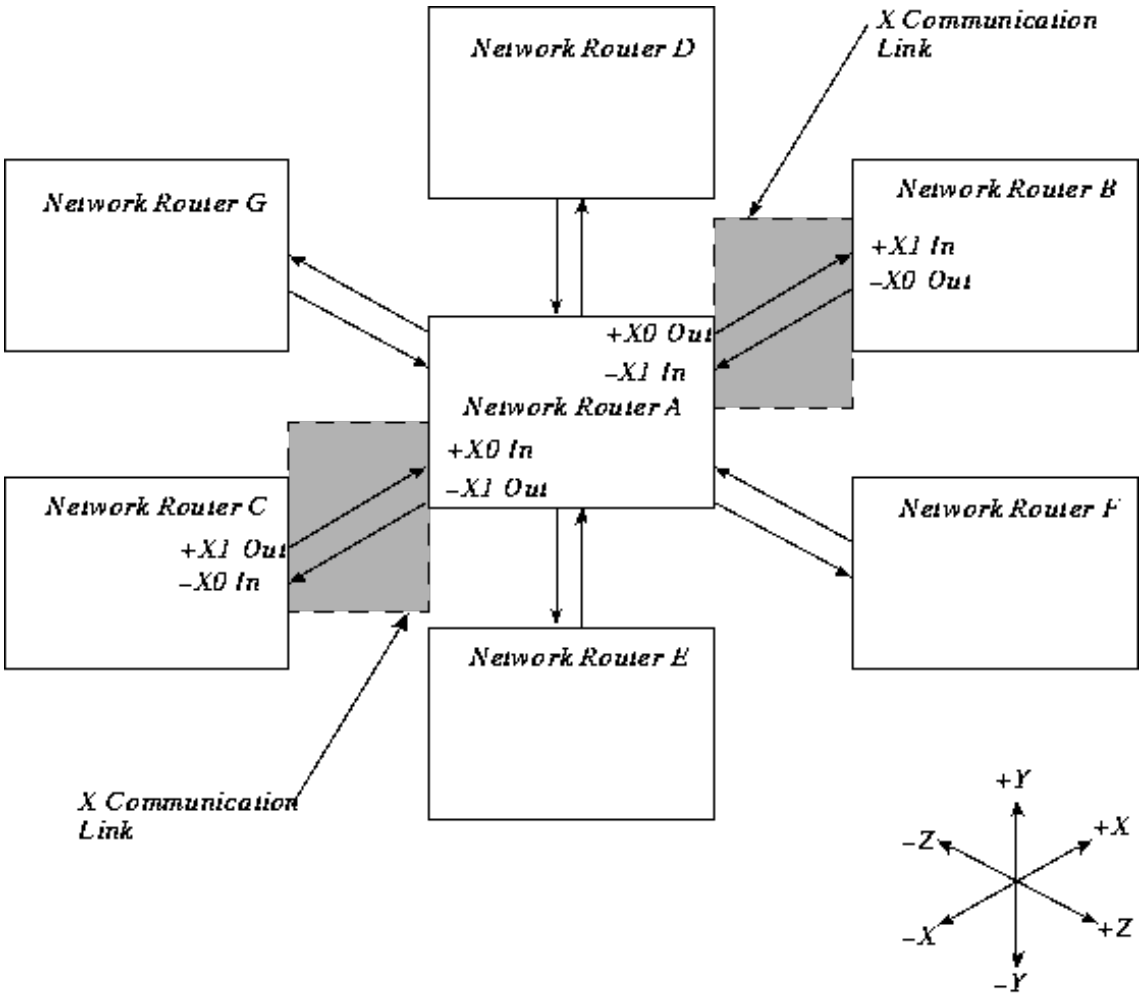
Table 1. Bits <2 : 0> of the R_ORIENT Register

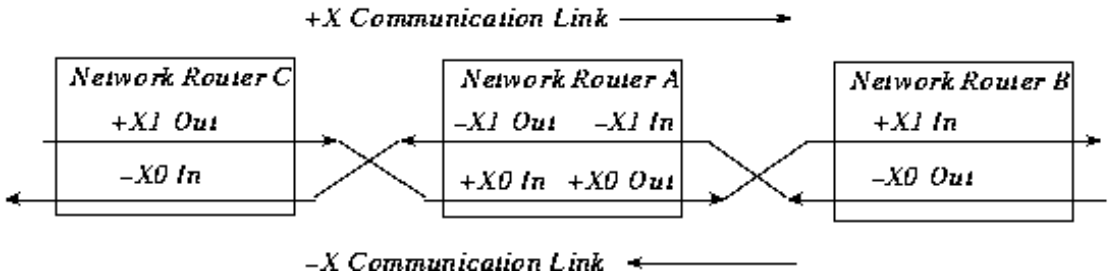
Bits	Value	Description

0	0	X0 port is +X and X1 port is -X
	1	X0 port is -X and X1 port is +X
1	0	Y0 port is +Y and Y1 port is -Y
	1	Y0 port is -Y and Y1 port is +Y
2	0	Z0 port is +Z and Z1 port is -Z
	1	Z0 port is -Z and Z1 port is +Z

In addition to the network router ports, the network router contains a network port, a PE port, and two I/O ports (refer again to Figure 3). The network port is a unidirectional channel that connects the support circuitry to a network router. The PE port is a unidirectional channel that connects the network router to the support circuitry. The I/O ports are unidirectional channels that connect the network router to the I/O circuitry.

Figure 4. Positive and Negative Communication Links





R Option

The R option contains the logic that makes up the network router. The R option contains the following components (refer to Figure 5):

- I Interface - The I interface buffers packet information for the input/output option, retrieves the packet's routing tag from a R_NET_LUT register, and generates control signals that identify which network port the I/O packet will use.

- C Interface - The C interface buffers packet information for the control option, retrieves routing tags from the R_NET_LUT registers, and generates control signals that identify which network port the packet will use.

- R Interface - The R interface contains the six network router ports that handle incoming and outgoing

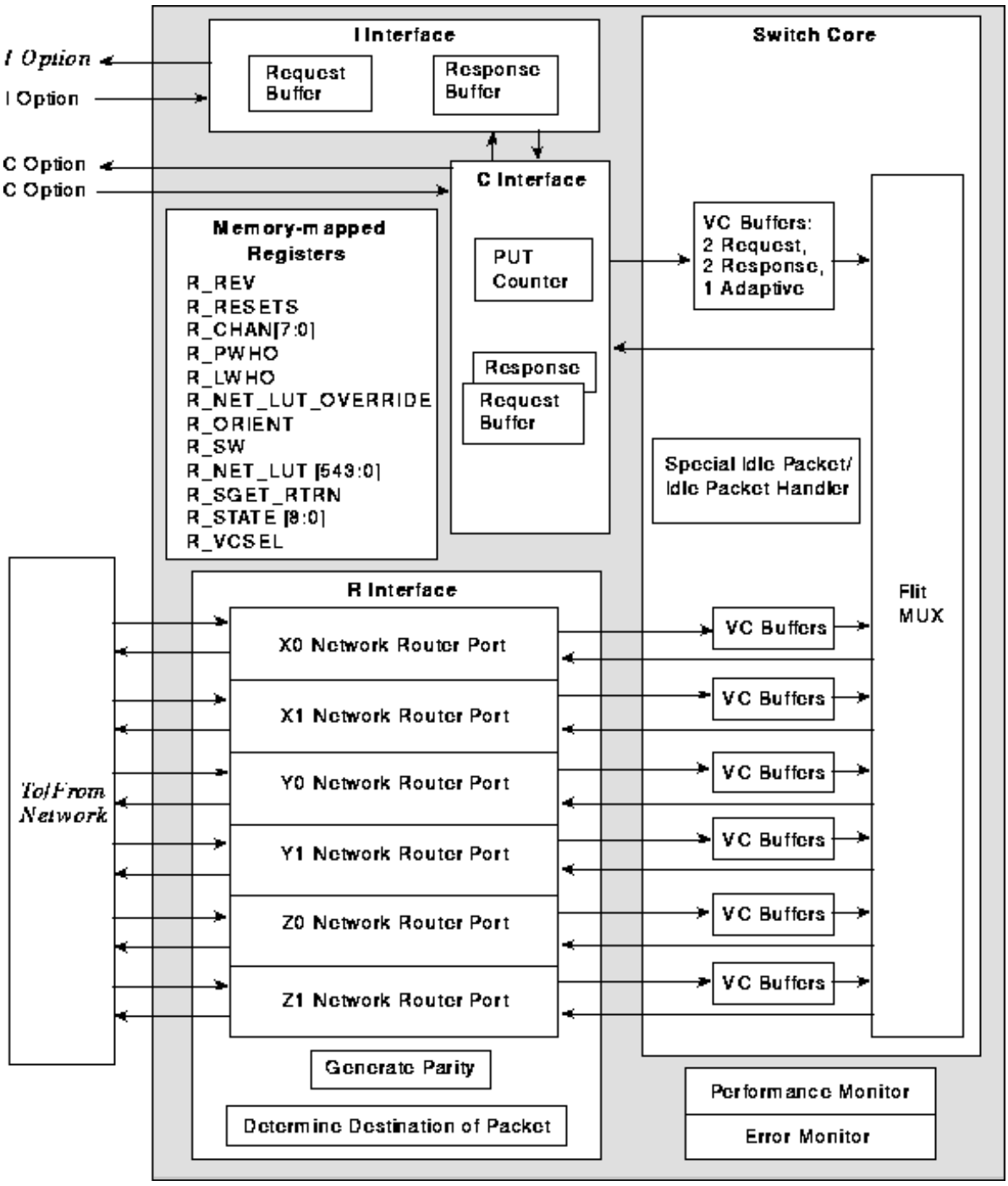
packets. For outgoing packets, the R interface generates parity and steers the packet out onto the interconnect network. For incoming packets, the R interface determines whether the packet can use the fast path and checks the parity bits and single-error correction bits for errors.

NOTE: The fast path is a path that bypasses the switch core. When no conflicts exist, the packet can use this path.

- Switch Core - The switch core buffers packet information for the I interface, the C interface, and the R interface. The switch core uses the control signals from the I interface, C interface, and R interface to determine where (which network router port) to steer the packet, then arbitrates for the network router port, and when the packet has priority, steers the packet to the appropriate network router port.

The switch core buffers the packet information in virtual channel (VC) buffers. There are five VC buffers: two request VC buffers (VC 0 and VC 2), two response VC buffers (VC 1 and VC 3), and one adaptive VC buffer for each network router port. The C interface uses VC0 and VC1 and the I interface uses VC2 and VC3.

Figure 5. R Option



The VC buffers prevent two types of communication deadlock conditions in the interconnect network. The following paragraphs describe these conditions.

A communication deadlock condition may occur if two nodes simultaneously transfer request or response information to each other. The CRAY SSD-T90 device uses two types of VC buffers to prevent this condition: request buffers and response buffers. These buffers provide separate destination buffers for request and response information.

A communication deadlock condition may also occur if all nodes in one dimension send request or response information to the next node in the dimension at the same time. For example, without the VC buffers, a deadlock condition may occur if all of the nodes in the X dimension send request information to the next node in the +X direction at the same time.

The VC buffers 0 through 3 are used in conjunction with datelines (software assigns the datelines) to eliminate channel

deadlock conditions. For example, when a request packet that is using VC 0 passes through a dateline node and does not switch to a new dimension, the dateline node buffers the packet in VC 2 rather than VC 0. The request packet continues to use VC 2 until the packet reaches the destination node.

NOTE: An error occurs when a packet uses VC 2 or VC 3 and the packet passes through a dateline node.

For example, Figure 6 shows four nodes in the X dimension. Node 000000 is transferring request information to the node that is two nodes away in the +X direction (Node 000002). The dateline communication link is the communication link that connects nodes 000001 and 000002.

Figure 6. X-dimension Virtual Channel Buffers

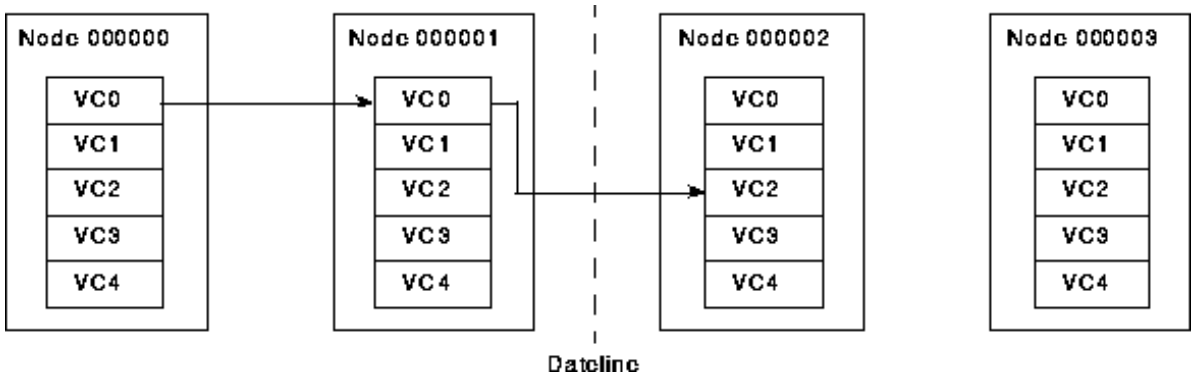
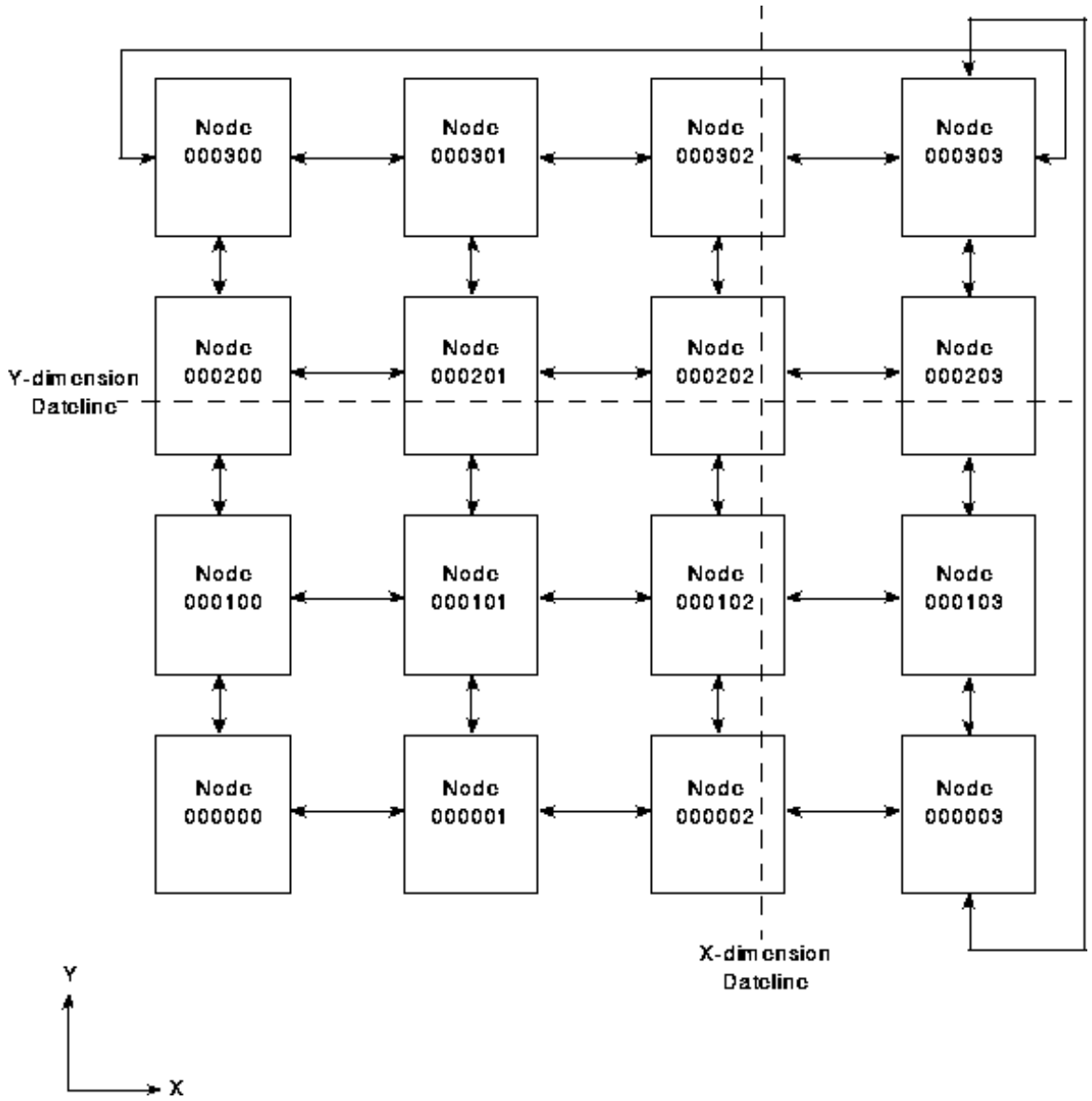


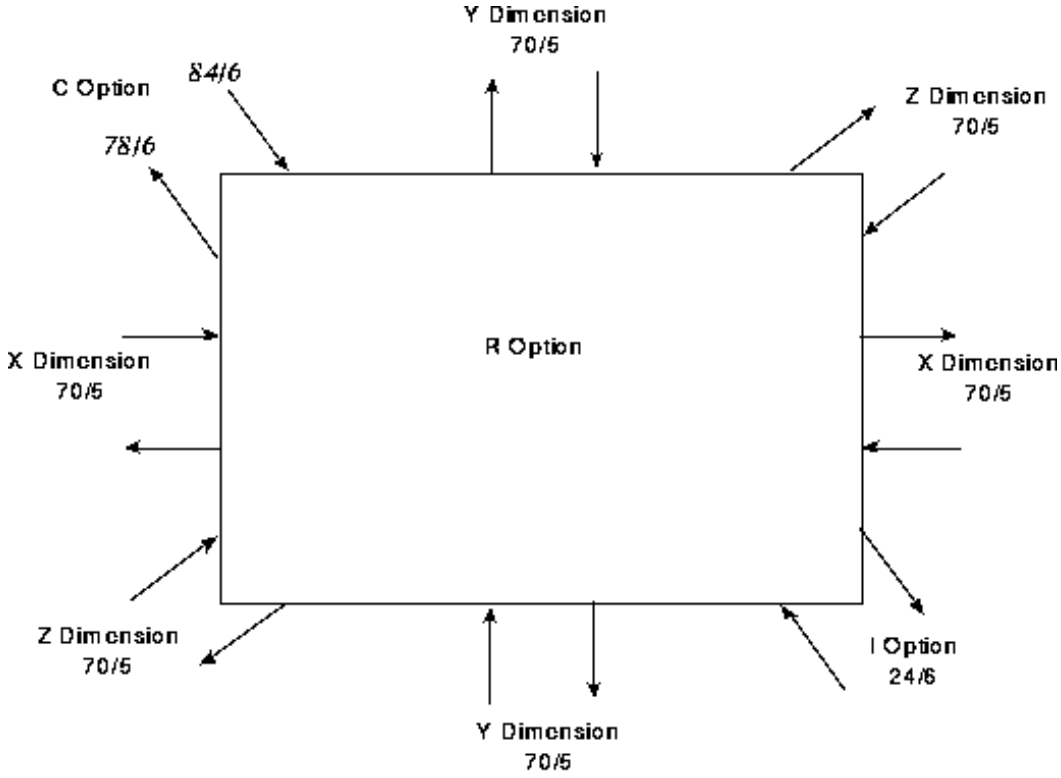
Figure 7 shows the X dimension dateline and the Y dimension dateline in a two-dimensional plane.

Figure 7. Dateline in a Two-dimensional Plane



The R option receives and transfers data over time-multiplexed channels. The time-multiplexed channels that connect the R options are 14-bit channels that make several transfers of data within one system clock period. The R option transfers 70 bits of data (in five 14-bit transfers) over this channel in one system clock period. The time-multiplexed channels that connect an R option to a C option are a 13-bit channel (output from the R option) and a 14-bit channel (output from the C option) (refer to Figure 8). The channel between the I option and the R option is 4 bits.

Figure 8. Time-multiplexed Channels on the R Option



NOTE: The x/y notational convention indicates the number of bits to be transferred on the channel in one system clock period (x) and how many transfers are required to transfer x number of bits in one system clock period (y).

The R option uses the packet formats shown in Table 2to send requests and responses to the C option. The R option uses the packet formats shown in Table 3 and Table 4 to send requests and responses to another R option. The R option uses the packet formats shown in Table 5 to send request and responses to the I option. Figure 9 describes the R option-to-R option flit formats.

Table 2. R Option-to-C Option Packet Formats

Command	Flit 0	Flit 1	Flit 2	Flit 3	Flit 4	Flit 5	Flit 6	Flit 7	Flit 8
Request									
DGET	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
DPUT	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND	Head	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
PUT4, PUT8	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PUTV4	Head	Words 0, 1	Words 2, 3	Words 4, 5	Words 6, 7	N/A	N/A	N/A	N/A
PUTV8	Head	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
GET	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GET_INC	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

GET_ADD	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SWAP	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
CSWAP	Head	Comp- erand	Swap- erand	N/A	N/A	N/A	N/A	N/A	N/A
GET_SET	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GET_CLEAR	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Response									
SEND	Head	Flit 1 contains hardware and software error information.							
SEND NACK/ Acknowledge	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PUT, DPUT	Head	Flit 1 contains software error information.							
PUT, DPUT Acknowledge	No response packet for the PUT acknowledge and the DPUT acknowledge; however, there is a PUT counter = 0 bit embedded in one of the two flits of the channel.								
GETV4	Head	Words 0, 1	Words 2, 3	Words 4, 5	Words 6, 7	N/A	N/A	N/A	N/A
GETV8	Head	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7

Table 3. R Option-to-R Option Request Packet Formats

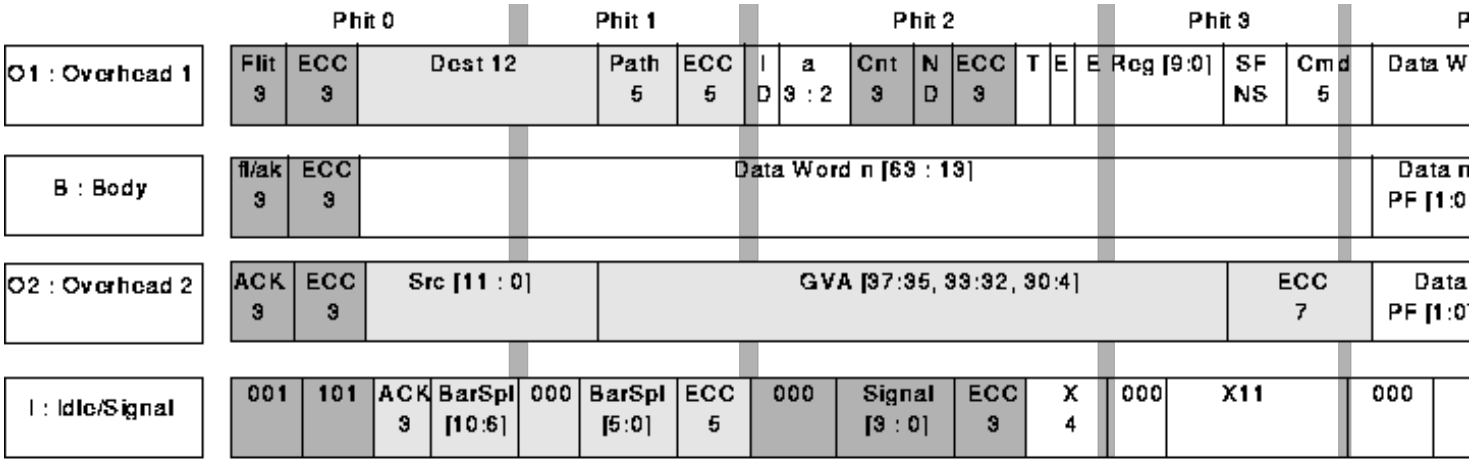
Command	Flit 0	Flit 1	Flit 2	Flit 3	Flit 4	Flit 5	Flit 6	Flit 7	Flit 8	Flit 9
DGET	O1	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
DPUT	O1	Wd 0 B	Wd1 B	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A
SEND	O1	Wd 0 B	Wd 1 B	Wd 2 B	Wd 3 B(ecc)	O2	Wd 4 B	Wd 5 B	Wd 6 B	Wd 7 B(ecc)
PUT	O1	Wd 0 B	Wd1 B	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A
PUTV4	O1	Wd 0 B	Wd 1 B	Wd 2 B	Wd 3 B(ecc)	O2 (ecc)	N/A	N/A	N/A	N/A
PUTV8	O1	Wd 0 B	Wd 1 B	Wd 2 B	Wd 3 B(ecc)	O2	Wd 4 B	Wd 5 B	Wd 6 B	Wd 7 B(ecc)
GET	O1	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GET_INC	O1	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

GET_ADD	O1	Wd 0 B	Wd1 B	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A
SWAP	O1	Comp- erand B	Swap- erand B	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A

Table 4. R Option-to-R Option Response Packet Formats

Command	Flit 0	Flit 1	Flit 2	Flit 3	Flit 4	Flit 5	Flit 6	Flit 7	Flit 8	Flit 9
GET, DGET	O1	Wd 0 B(ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PUT, DPUT, Atomic Operation Error	O1	Wd 0 B (ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PUT, DPUT	O1	O2 (ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GETV4	O1	Wd 0 B	Wd 1 B	Wd 2 B	Wd 3 B(ecc)	O2 (ecc)	N/A	N/A	N/A	N/A
GETV8	O1	Wd 0 B	Wd 1 B	Wd 2 B	Wd 3 B(ecc)	O2	Wd 4 B	Wd 5 B	Wd 6 B	Wd 7 B(ecc)
Atomic Operation	O1	Wd 0 B(ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND Error	O1	Wd 0 B(ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND NACK	O1	Wd 0 B(ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND ACK	O1	Wd 0 B(ecc)	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Figure 9. R Option-to-R Option Flit Formats (70 Bits)



T is the performance monitor tag bit that is used to tag packets that are being measured. This tag is located i of the packet, except for a PUT request. For the PUT request, the T tag is located in the first body.

ND is the network destination bit. A 1 indicates that the destination is either an R option or an I option.

The a3 : 2 in the O1 flit is global virtual address (GVA) bits [3 : 2]. GVA bit [2] is needed for unaligned GETV4 ID is GVA bit [34] for requests and is equivalent to ND for responses.

E is E register 10. This bit is listed separately to indicate that it must be sent (even on PUT commands).

The EReg [9 :0] field in the O1 flit are included (even on PUT responses) so that the I option can determine w commands have responded.

On requests, the payload flags (PFs) have the upper bit set for a parity error on the upper 32 bits of the payloa lower bit is set for a parity error on the lower 32 bits of the payload words. For responses, the flags are the da

The Cnt field indicates how many more double flits (after the header) need to be counted before the end of the example, the Cnt field contains a 1 for a 4-flit packet and a 4 for a 10-flit packet.

Sig[3] = x, Sig[2] = next flit is an idle flit, Sig[1] = channel bit, Sig[0] = heartbeat.

The 4-bit ECC field protects the 6 bits just to its left, giving a (10,6) code.

Table 5. R Option-to-I Option Packet Formats

Command	Flit 0	Flit 1	Flit 2	Flit 3	Flit 4	Flit 5	Flit 6	Flit 7	Flit 8
Request									
SEND	Head	Wd 0	Wd 1	Wd 2	Wd 3	Wd 4	Wd 5	Wd 6	Wd 7
PUT8	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GET8	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
Response									
GET8,DGET	Head	Data or Error	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND Error	Head	Error	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND NACK	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A
SEND ACK	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
PUT Error	Head	Error	N/A	N/A	N/A	N/A	N/A	N/A	N/A

PUT8, PUTV8	Head	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A
GETV8	Head	Wd 0	Wd 1	Wd 2	Wd 3	Wd 4	Wd 5	Wd 6	Wd 7
SGET	Head	Body	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Routing Tag Look-up Table

Each network router has a set of network router look-up table (R_NET_LUT) registers. When software that is running in the microprocessor signals the support circuitry to transfer information to a remote PE, the network router uses the logical PE number to reference an R_NET_LUT register; the network router reads a routing tag from this register.

The routing tag determines the path that a packet follows when it travels from the source physical node to the destination physical node. The routing tag consists of the following fields:

- Initial +X, +Y, or +Z hop
- X-dimension address and direction
- Y-dimension address and direction
- Z-dimension address and direction
- Final -Z hop
- Adaptive routing bit

More information about each field is provided later in this document.

There are a total of 544 R_NET_LUT registers. For systems with 544 or fewer PEs, software assigns one R_NET_LUT register to each logical PE in the system. For larger systems, software assigns one R_NET_LUT register to 4 logical PEs in the system.

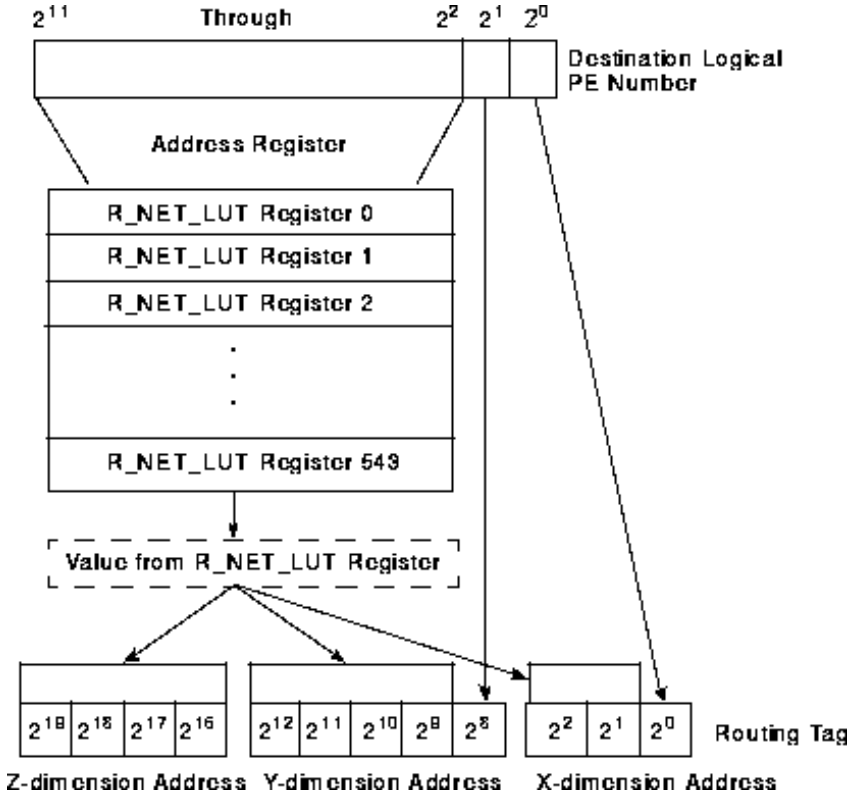
Software uses the R_LUT_OVERRIDE register to define how the R_NET_LUT registers are assigned for a system. The R_LUT_OVERRIDE is a memory-mapped register that specifies 1 of 4 modes that the network router can use while creating a routing tag (refer to Table 6).

Table 6. Modes of the R_LUT_OVERRIDE Register

Mode	Description
0	Each R_NET_LUT register corresponds to one logical PE number. (Not valid for systems with more than 544 PEs.)
1	The support circuitry replaces bit 0 of the routing tag with bit 0 of the logical PE number and replaces bit 8 of the routing tag with bit 1 of the logical PE number (refer to Figure 10).

2	The support circuitry replaces bit 0 of the routing tag with bit 1 of the logical PE number and replaces bit 8 of the routing tag with bit 0 of the logical PE number.
3	The support circuitry uses bits <37 : 0> of the R_LUT_OVERRIDE register instead of the R_NET_LUT registers.

Figure 10. Creating a Routing Tag Using Mode 1

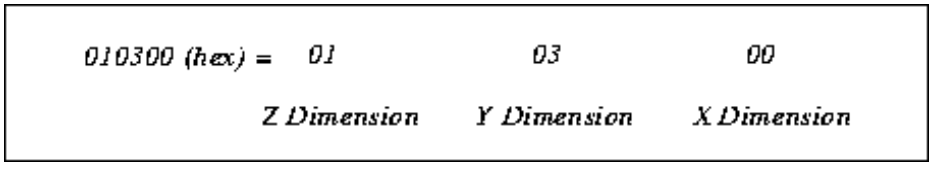


Node Identification

Software identifies each node within the three-dimensional matrix by a physical PE number and a logical PE number. Software also identifies a node by a virtual PE number when it assigns a PE to a partition.

A physical PE number is a hexadecimal number that indicates the physical position of the node in relation to a node that software designates as the origin node, or 000000 (refer to Figure 12). For example, the PE with the physical PE number 010300 (hex) is physically positioned 3 hops in the +Y dimension from the origin node and 1 hop in the +Z dimension (refer to Figure 11).

Figure 11. Dimension Assignment for the Physical PE Number



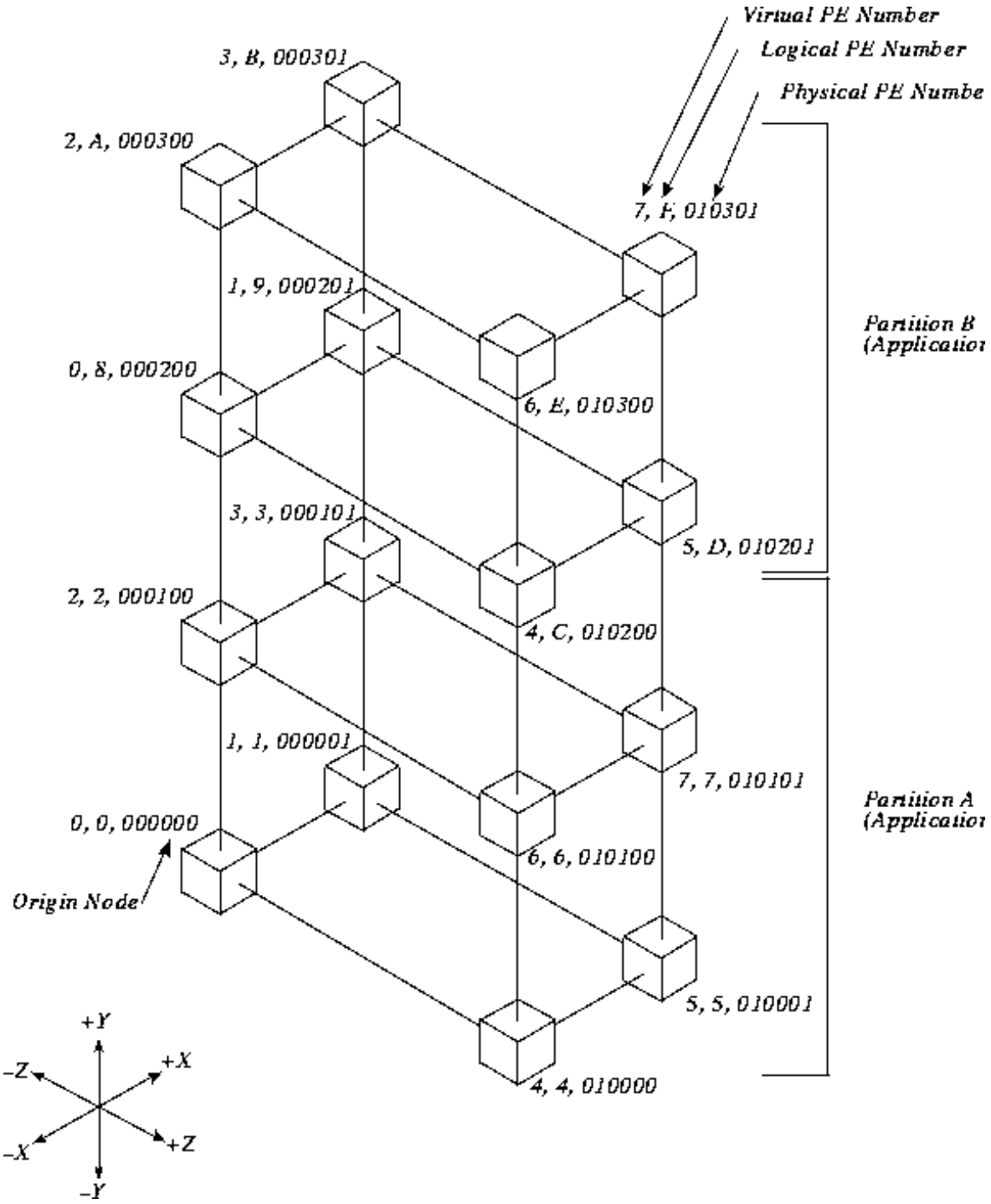
A logical PE number indicates how software distributes the node in the interconnect network. For example, you can

logically remove a bad node from the CRAY SSD-T90 device by assigning the logical number of the bad node to a different physical node.

Software also assigns a virtual PE number to a node. The virtual PE number identifies the node within a partition. A partition is a group of PEs that are assigned to one application. For example, in Figure 12 partition B contains logical PEs 8 through F (hex). For this partition, software assigns virtual PE numbers 0 through 7 to logical PEs 8 through F, respectively.

NOTE: Software always assigns virtual PE number 0 to the node with the smallest logical PE number in the partition. The logical PE with the next smallest number becomes virtual PE 1, and so on.

Figure 12. Logical, Virtual, and Physical Node Numbers for a 16-PE Device



The 16 PEs shown in Figure 12 reside on 4 processing element printed circuit boards (PCBs). The communication links are made up of foil traces on the PCB (internal communication links) or a combination of foil traces, edge connectors, and ribbon cables (external communication links). Refer to Figure 13.

Figure 13. External and Internal Communication Links of a PCB

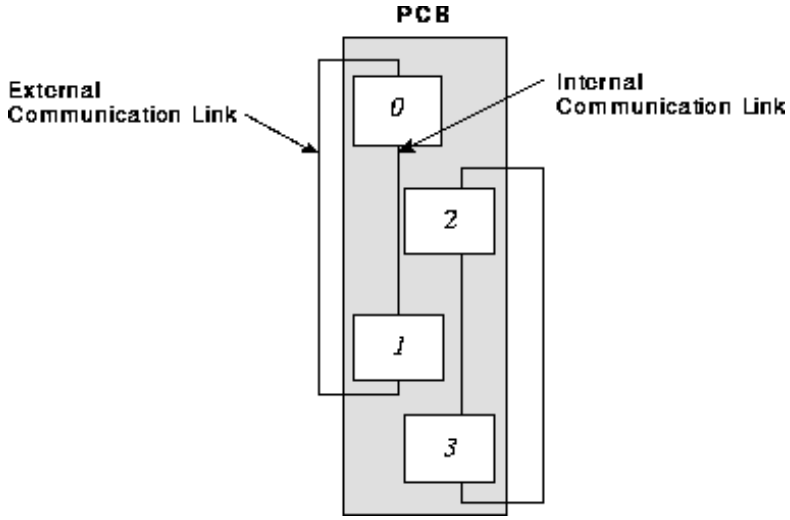
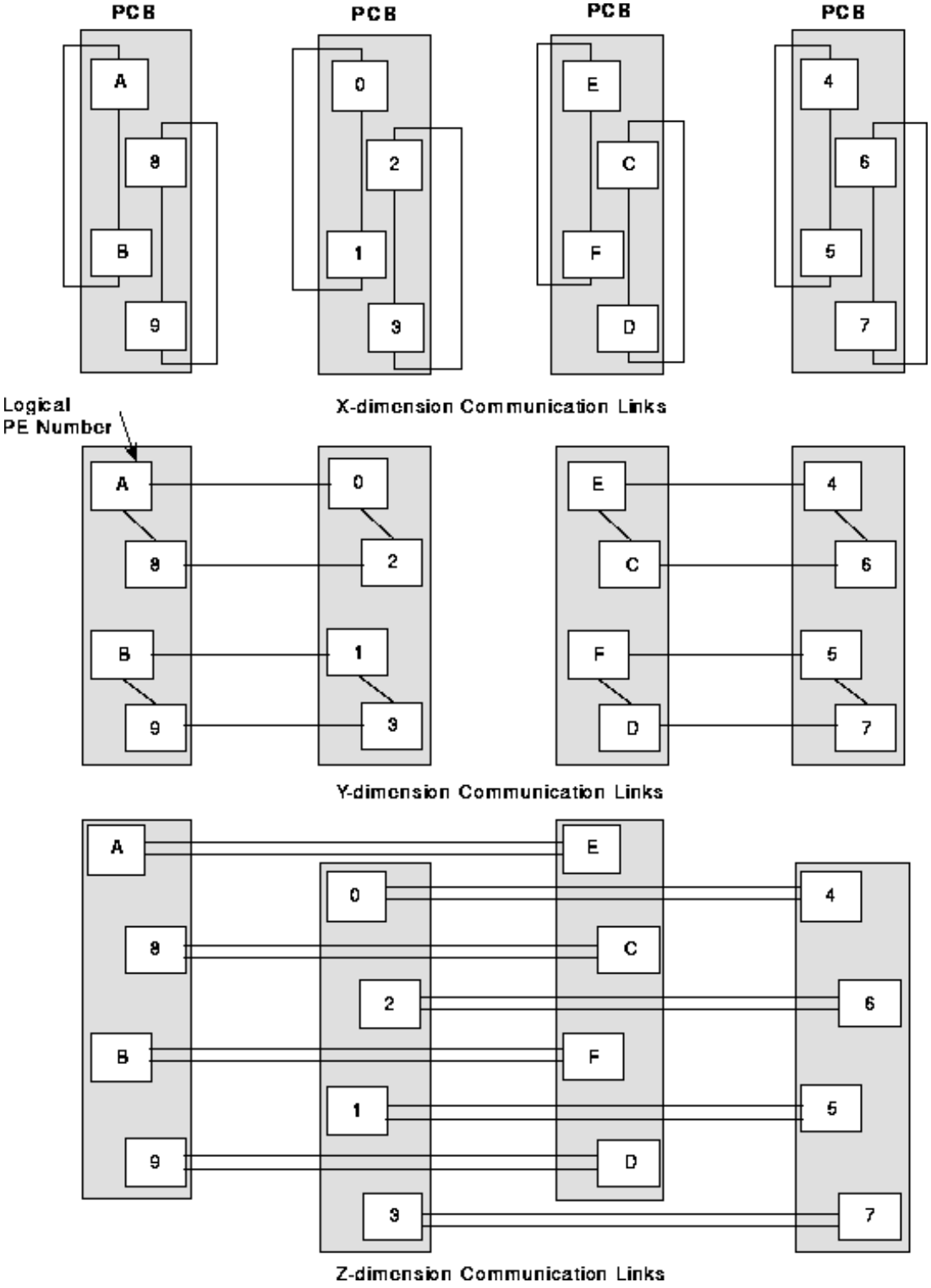


Figure 14 shows the X, Y, and Z communication links between the 4 PCBs in a 16-PE device. In this example, the X-dimension communication links do not connect to another PCB; however, 2 of the 4 communication links on each PCB are external. In other words, they use edge connectors to route signals. The Y-dimension communication links also have internal and external communication links, but here the external communication links connect two different PCBs. The Z-dimension communication links are always external communication links that connect at least two PCBs.

Figure 14. X, Y, and Z-dimension Communication Links for a 16-PE Device



Types of Information Routing

There are three types of information routing: special routing, direction order routing, and adaptive routing.

Special Routing

PEs cannot use the interconnect network until software programs the network memory-mapped registers. To program

these registers, software must use a special routing technique. This special routing technique uses six delta values to route packets of information through the interconnect network. These values are:

1. Primary delta X value
2. Primary delta Y value
3. Primary delta Z value
4. Secondary delta X value
5. Secondary delta Y value
6. Secondary delta Z value

When information follows a special routing path, it travels in the order shown in the list above.

The delta values indicate the number of hops (negative or positive) that the packet will make in each dimension. For example, when the primary delta X value is set to +3, the information completes 3 hops in the positive X direction.

Special routing also transfers packets of information through a nonblocking network. A nonblocking network is free of potential deadlock conditions; however, if two packets of information access the same network router port at the same time, one packet overwrites the other.

The following text describes how the microprocessor uses special routing to write to a network router register. The step numbers correspond to the numbers in Figure 15.

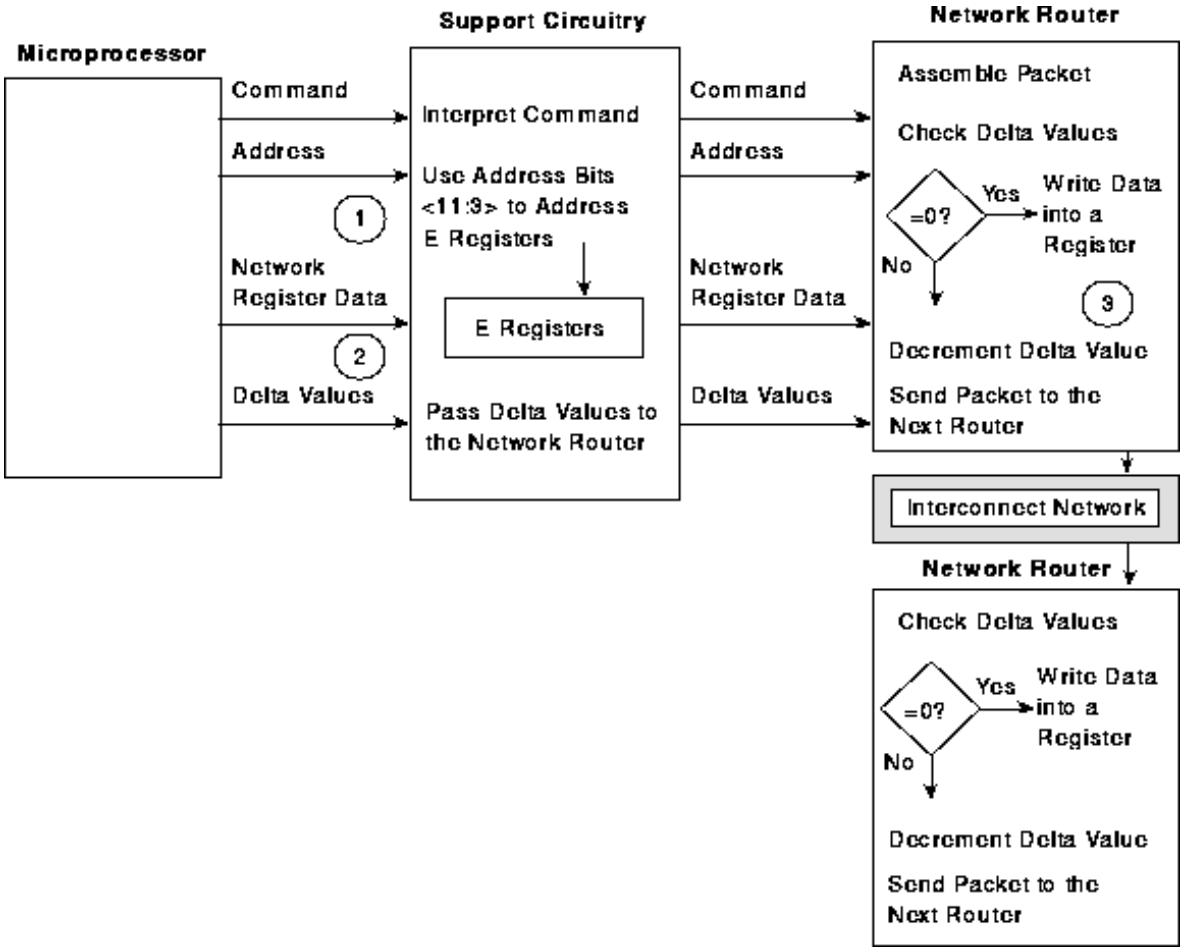
1. Software issues a STORE command. The STORE command signals the support circuitry to transfer network register data from the microprocessor to an E register. The microprocessor also sends the data and an address to the support circuitry. Address bits 3 through 11 indicate the E register. Address bits 0 through 2 are set to 0's. Address bit 24 indicates the user or system E-register context.

2. Software issues a special put (SPUT) E-register command. The SPUT command signals the support circuitry to transfer the network register data from the E register to a network router register that is designated by the microprocessor-supplied delta values.

3. The network router uses the delta values to determine whether the register data should be written into one of its registers. When the delta values are 0, the network router writes the network register data into the appropriate register. When the delta values are not 0, the network router decrements the appropriate delta value and sends the packet to the next network router.

   The network routers continue to pass the network router data until it reaches the destination network router (primary and secondary delta values = 0).

   Figure 15. Special Routing Block Diagram for a Write Operation



The following text describes how the microprocessor uses special routing to read from a network router register. The step numbers correspond to the numbers in Figure 16.

1. Software issues a special GET command (SGET). The SGET command signals the support circuitry to transfer a 64-bit word of data from a network router register (designated by the microprocessor-supplied delta values) to a destination E register.

2. The local network router uses the delta values to determine whether the register data should be read from one of its registers. When the delta values are 0, the network router reads the network register data from the appropriate register and sends the data to the support circuitry. When the delta values are not 0, the network router decrements the appropriate delta value and sends a packet that contains the SGET command to the next network router.

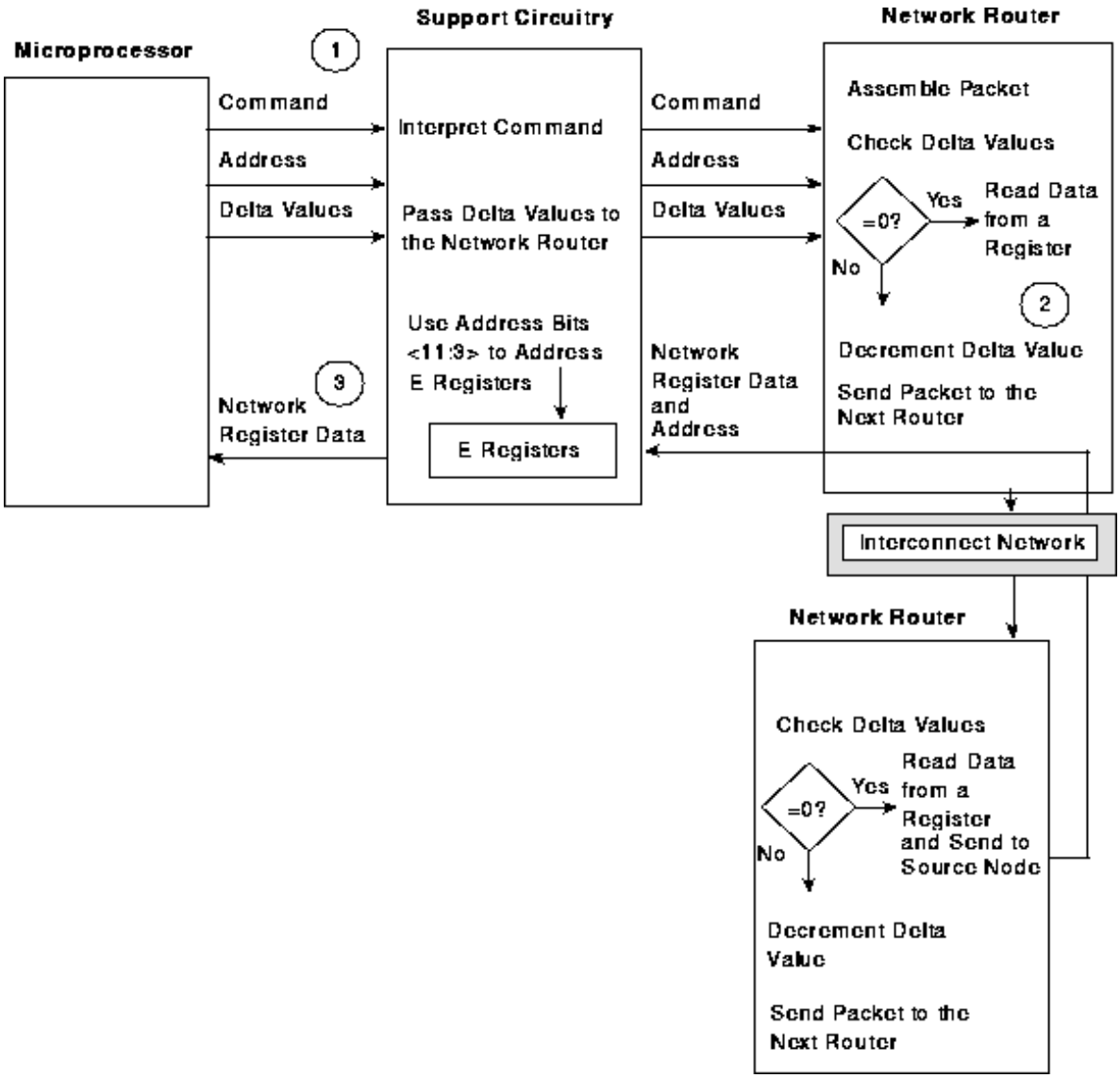
   The network routers continue to pass the network router data until it reaches the destination network router (delta values = 0). When the destination network router receives the SGET command, it reads the data from the appropriate network router register, places the data in a packet, and sends the packet back to the source node.

   NOTE: To return the SGET response packet to the source node, the network router uses the delta values from the R_SGET_RTRN register. Software wrote these delta values into this register before issuing the SGET command.

   When the source node receives the packet, the network router passes the network router data to the support circuitry. The support circuitry writes the data into an E register. The address of the E register was supplied by the microprocessor when it issued the SGET command.

3. Software issues a LOAD command. The LOAD command signals the support circuitry to transfer the network register data from the E register to the microprocessor.

Figure 16. Special Routing Block Diagram for a Read Operation



Direction Order Routing

Software uses direction order routing to transfer data through the interconnect network in the following order:

1. Any initial hop in the +X, +Y, or +Z direction
2. Any travel in the +X direction
3. Any travel in the +Y direction
4. Any travel in the +Z direction

5. Any travel in the -X direction
6. Any travel in the -Y direction
7. Any travel in the -Z direction
8. Any final hop in the -Z direction

A routing tag, like the physical PE number, indicates the physical position of the destination node with respect to the origin node. The routing tag consists of the fields listed in Table 7.

The network routers use the routing tag to determine when a packet should continue traveling in the current dimension, switch to the next dimension, or be sent to the support circuitry. To make this determination, the network routers compare the routing tag to their physical PE numbers. When a dimension address of the routing tag is not equal to the dimension address of the physical PE, the network router continues to transfer the packet in the current dimension. When a dimension address of the routing tag is equal to the dimension address of the physical PE, the network router switches the packet to the next dimension. Once the packet reaches the destination PE (the routing tag and physical PE number are equal), the network router sends the packet to the support circuitry.

NOTE: The offline diagnostic interface refers to direction order routing as deterministic routing.

Table 7. Fields of the Routing Tag

Bits	Fields	Description
<2 : 0>	X-dimension address	This field indicates the X-dimension physical address of the destination node.
<6 : 3>	Not applicable	These bits are not used.
7	X-dimension direction	This field indicates the direction the packet will travel in the X dimension. 0 = +X, 1= -X
<12 : 8>	Y-dimension address	This field indicates the Y-dimension physical address of the destination node.
<14 : 13>	Not applicable	These bits are not used.
15	Y-dimension direction	This field indicates the direction the packet will travel in the Y dimension. 0 = +Y, 1= -Y
<19 : 16>	Z-dimension address	This field indicates the Z-dimension physical address of the destination node.
<22 : 20>	Not applicable	These bits are not used.
23	Z-dimension direction	This field indicates the direction the packet will travel in the Z dimension. 0 = +Z, 1= -Z

<31 : 24>	Not applicable	These bits are not used.
<33 : 32>	Initial +X, +Y, +Z hop	This field indicates whether a packet makes an initial hop in the +X, +Y, or +Z direction before continuing on the path that is determined by the routing tag. 00 = +X, 01 = +Y, 10 = +Z, 11 = No initial +hop
<35 : 34>	Not applicable	These bits are not used.
36	Final -Z hop	The final -Z hop field indicates whether a packet makes a final hop in the -Z direction after completing the path determined by the routing tag. 0 = No final hop in the -Z direction, 1 = Final hop in the -Z direction
37	Adaptive routing	This bit indicates the routing method. 0 = Direction order routing, 1 = Adaptive routing
<63 : 38>	Not applicable	These bits are not used.

Example of Direction Order Routing

Figure 17 shows an example in which logical PE A (LPE A) sends a packet to LPE 7. To do this, LPE A uses the destination logical PE number 7 to address the routing tag look-up table. From this table, LPE A retrieves a routing tag that consists of the following information:

- No initial hop (INITIAL_HOP = 3)
- X direction = + (X_SGN = 0)
- X address = 01 (X_ADR = 01)
- Y direction = + (Y_SGN = 0)
- Y address = 01 (Y_ADR = 01)
- Z direction = + (Z_SGN = 0)
- Z address = 01 (Z_ADR = 01)
- No final hop (FINAL_HOP = 0)

The network router of LPE A uses this routing tag to determine the direction in which it will steer the packet out onto the interconnect network. Remember, for direction-order routing, the packet will travel to the destination in the following order: initial hop, +X, +Y, +Z, -X, -Y, -Z, and final hop.

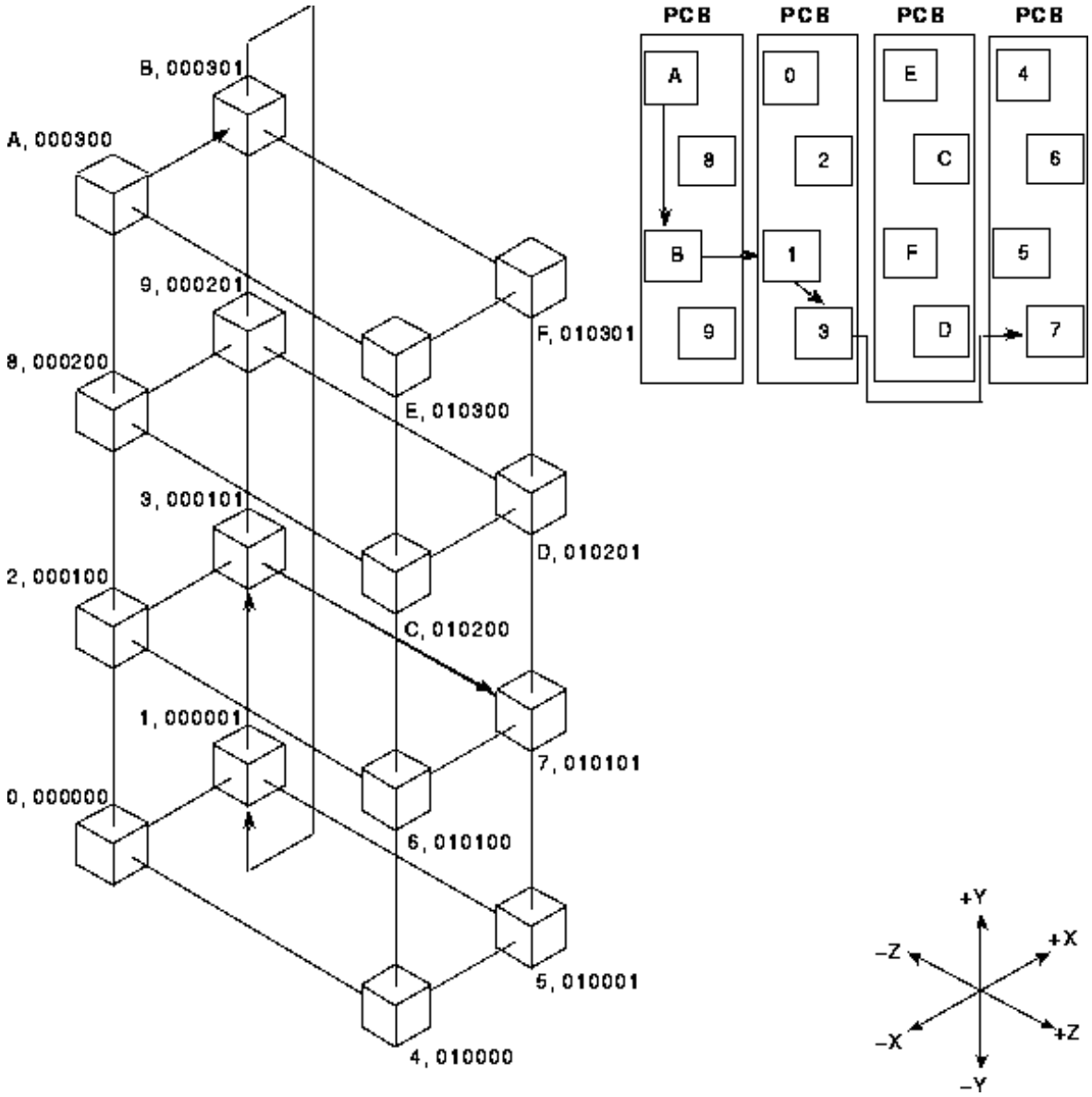
The network router of LPE A sends the packet out onto the interconnect network in the +X direction. When the packet reaches the next node, the network router of that node compares the X-dimension address of the routing tag to the X-dimension address of the physical PE number. The X-dimension address of the routing tag is 01 and the X-dimension address of the physical PE number is 01. Because the two X-dimension addresses are equal, the network router switches the packet to the next dimension.

The next positive dimension that the packet will travel is the +Y dimension. In this dimension, the packet will initially travel 1 hop in the positive direction. At this node, the network router compares the Y-dimension address of the routing tag to the Y-dimension address of the physical PE number. The Y-dimension address of the routing tag is 01, and the Y-dimension address of the physical PE number is 00. Because the two addresses are not equal, the network router continues to transfer the packet in the +Y dimension.

At the next node, the network router compares the Y-dimension address of the routing tag to the Y-dimension address of the physical PE number. The Y-dimension address of the routing tag is 01, and the Y-dimension address of the physical PE number is 01; therefore, the network router switches the packet to the next dimension.

The final dimension that the packet will travel is the +Z dimension. In the Z dimension, the packet travels 1 hop to the next node. At this node, the network router compares the Z-dimension address of the routing tag to the Z-dimension address of the physical PE number. The Z-dimension address of the routing tag is 01, and the Z-dimension address of the physical PE number is 01; therefore, the packet is at its destination. The network router then transfers the packet to the support circuitry.

Figure 17. Example of LPE A Sending a Packet to LPE 7



Example of Routing Using the Initial-hop Option

Software may use the initial-hop option to make routing easier for packets that travel to physical nodes that reside in a partial plane or to route around a bad communication link.

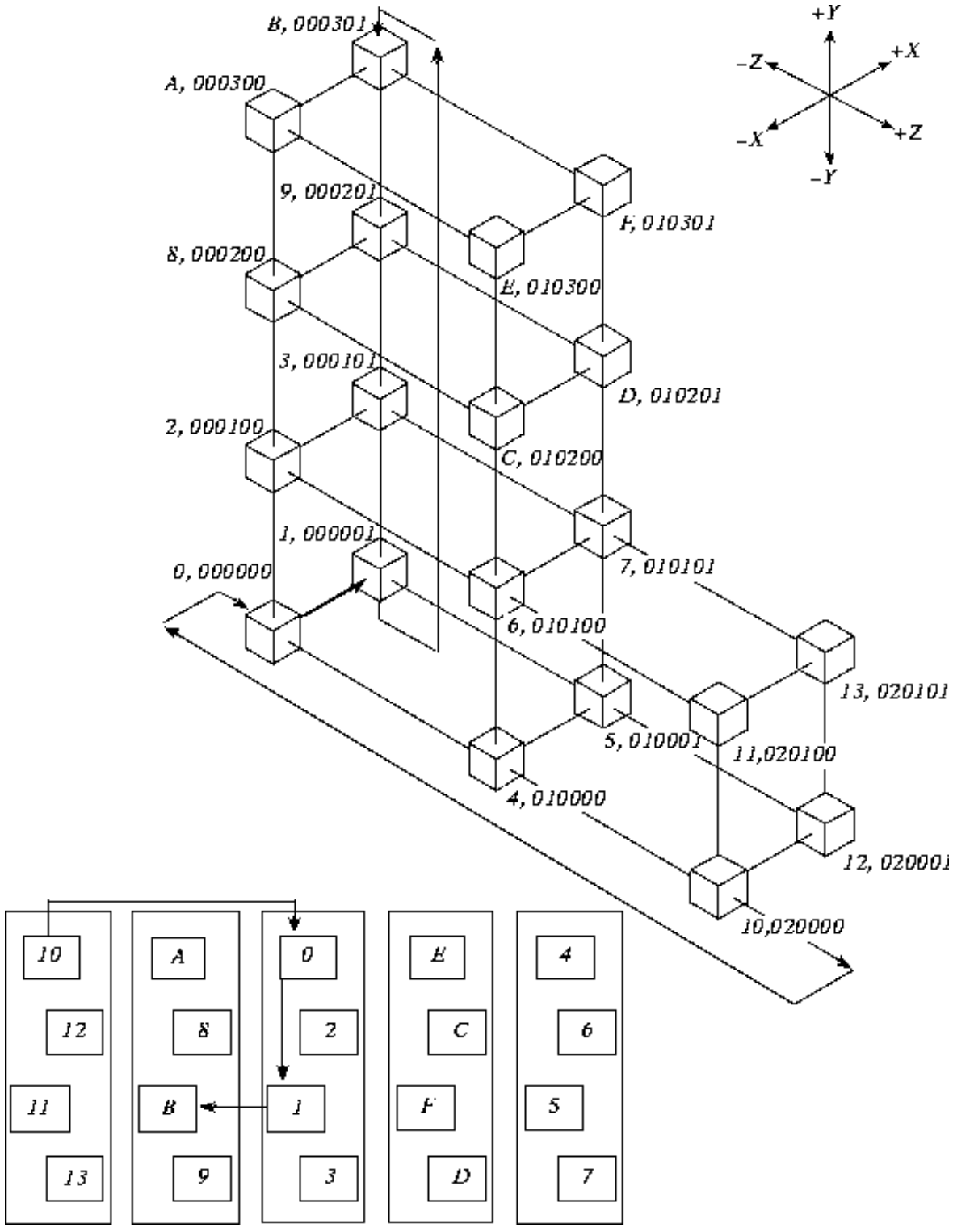
Figure 18 illustrates routing using the initial-hop option. In this example, LPE 10 sends a packet to LPE B. LPE 10 uses the destination logical PE number B to address a routing-tag look-up table. From this table, LPE 10 retrieves a routing tag that consists of the following information:

- Initial hop in the +Z dimension (INITIAL_HOP = 2)
- X direction = + (X_SGN = 0)
- X address = 01 (X_ADR = 01)
- Y direction = - (Y_SGN = 1)
- Y address = 03 (Y_ADR = 03)

- Z direction = + (Z_SGN = 0)
- Z address = 00 (Z_ADR = 00)
- No final hop (FINAL_HOP = 0)

Note that the initial hop field of the routing tag is a 2. This indicates that the packet will travel one hop in the +Z dimension before completing the path as determined by the rest of the routing tag. (For this example, after the packet completes the initial hop, it travels in the +X dimension until it reaches address 01, the +Z dimension until it reaches address 00, and the -Y dimension until it reaches address 03.)

Figure 18. Example of Routing Using the Initial-hop Option



Example of Routing Using the Final-hop Option

Like the initial-hop option, software may use the final-hop option to make routing easier for packets that travel to physical nodes that reside in a partial plane or to route around a bad communication link.

Figure 19 illustrates routing using the final-hop option. In this example, LPE F sends a packet to LPE 12. LPE F uses the destination logical PE number 12 to address a routing-tag look-up table. From this table, LPE F retrieves a routing tag that consists of the following information:

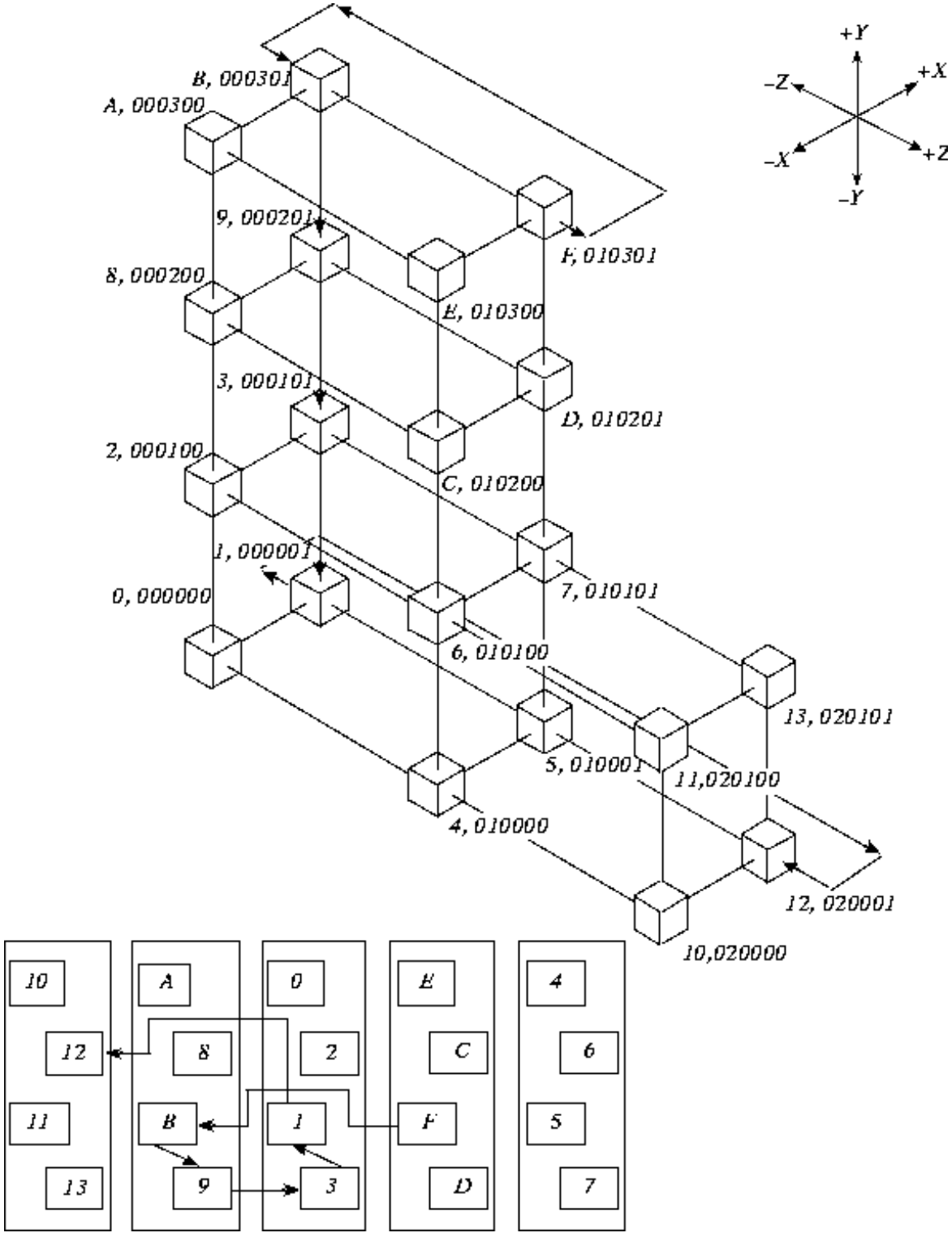
- No initial hop (INITIAL_HOP = 3)

- X direction = + (X_SGN = 0)
- X address = 01 (X_ADR = 01)
- Y direction = - (Y_SGN = 1)
- Y address = 00 (Y_ADR = 00)
- Z direction = + (Z_SGN = 0)
- Z address = 00 (Z_ADR = 00)
- Final hop in the -Z dimension (FINAL_HOP = 1)

Note that the final-hop field of the routing tag is 1. This 1 indicates that the packet will travel one hop in the -Z dimension after it completes the path determined by the rest of the routing tag. (For this example, before the packet completes the final hop, it travels in the +X dimension until it reaches address 01, the +Z dimension until it reaches address 00, and the -Y dimension until it reaches address 00.)

NOTE: When the Z_SGN bit is a 1, the hardware forces the final hop bit to 0 because a packet cannot travel in the -Z dimension and make a final hop.

Figure 19. Example of Routing Using the Final-hop Option



Adaptive Routing

Software uses adaptive routing to transfer data through the interconnect network in the following order:

1. Any travel in the +X direction (lowest)
2. Any travel in the +Y direction
3. Any travel in the +Z direction

4. Any travel in the -X direction
5. Any travel in the -Y direction
6. Any travel in the -Z direction (highest)

Software enables adaptive routing by setting the adaptive routing bit in the routing tag to a 1. Setting this bit to a 1 enables the network router to use either the adaptive VC of a higher dimension or the direction-order VC of a lower dimension.

For example, Figure 20 shows the adaptive routing path and the direction-ordered path from LPE 0 to LPE D. The routing tag for this packet contains the following information:

- X direction = + (X_SIGN = 0)
- X address = 01 (X_ADR = 01)
- Y direction = + (Y_SIGN = 0)
- Y Address = 02 (Y_ADR = 02)
- Z direction = + (Z_SIGN = 0)
- Z address = 01 (Z_ADR = 01)
- Adaptive bit = 1

From LPE 0, the network router requests both the direction-ordered VC of the +X dimension (lower dimension) and the adaptive VC of the +Z dimension (higher dimension).

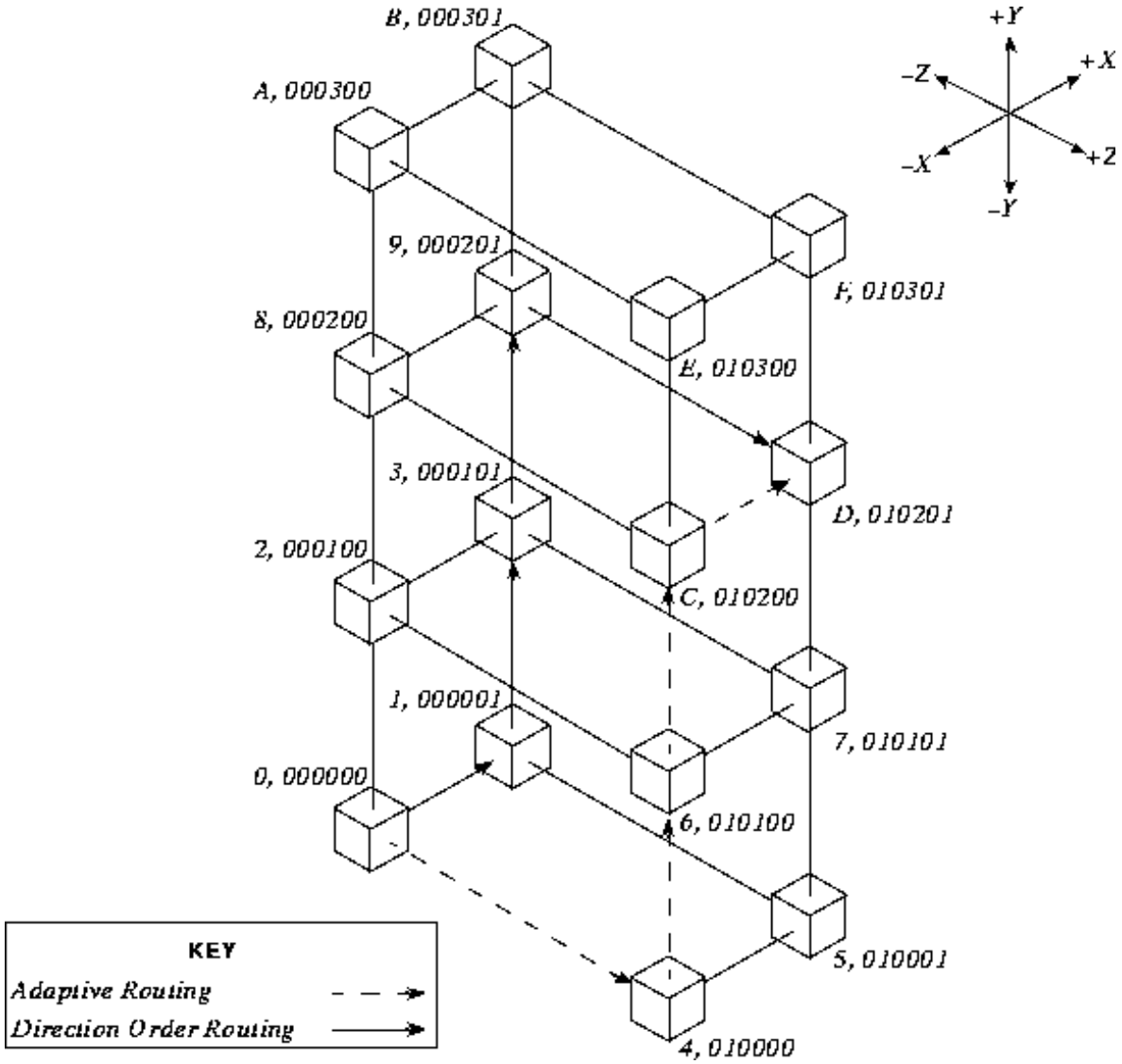
NOTE: When the direction-ordered VC and the adaptive VC are available at the same time, the packet uses the adaptive VC.

In this example, the packet follows the adaptive path; therefore, the packet travels in the +Z dimension. At LPE 4, the network router requests the direction-ordered VC of the +X dimension and the adaptive VC of the +Y dimension (next higher dimension). Following the adaptive path, the packet travels in the +Y dimension. At LPE 6, the network router requests the direction-ordered VC of the +X dimension and the adaptive VC of +Y dimension again (because the routing tag specifies address 02 in the +Y dimension). Following the adaptive path, the packet travels in the +Y dimension. At LPE C, the +Y dimension travel is complete; therefore, the network router requests the direction-ordered VC and the adaptive VC of the +X dimension.

NOTE: When the network router requests the direction-ordered VC and the adaptive VC of the same dimension and both VCs are available, the packet uses the direction-ordered VC.

For this example, the packet uses the adaptive VC and travels one hop in the +X dimension to the destination node.

Figure 20. Example of Adaptive Routing



Interconnect Network Errors

The R option logs interconnect network errors in the R_ERR0 memory-mapped register (refer to Table 8 and to Figure 21). Software uses the R_ERR1 memory-mapped register to enable the error interrupts.

R_ERR0

When a bit of the R_ERR0 register is set to 1, the bit indicates that the corresponding error has occurred.

Table 8. R_ERR0 Register Bit Layout

Bits	Description
<5 :	When set to 1, each of these bits indicates that a dateline violation occurred for the corresponding port. 0 = X0 port 3 = Y1 port

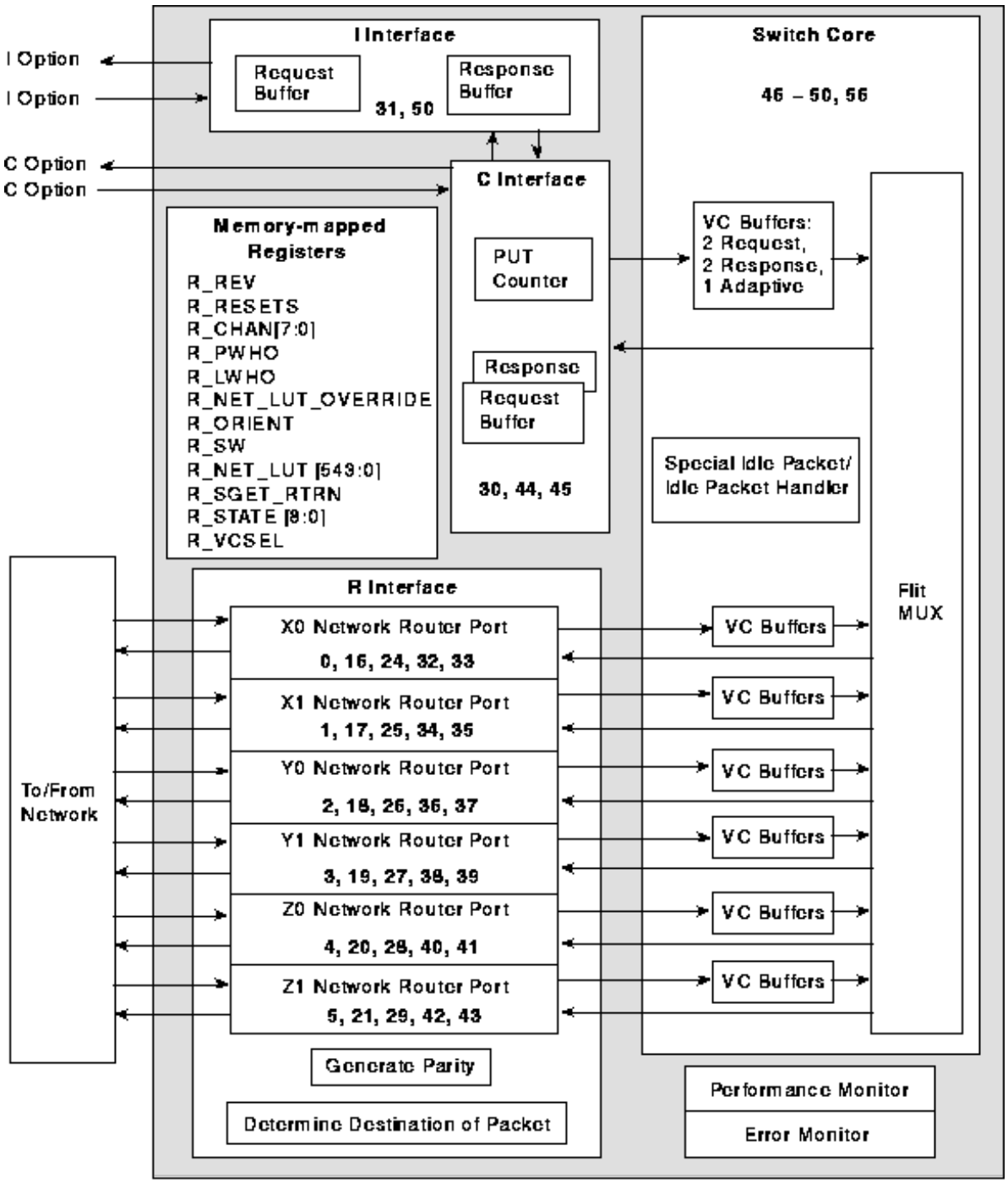
0>	1 = X1 port 4 = Z0 port 2 = Y0 port 5 = Z1 port
<7 : 6>	These bits are reserved.
<13 : 8>	These bits are not used.
<15 : 14>	These bits are reserved.
<21 : 16>	When set to 1, each of these bits indicates that a parity error occurred on a head flit type and this caused the packet to be accepted, then ignored on the corresponding port. 16 = X0 port 19 = Y1 port 17 = X1 port 20 = Z0 port 18 = Y0 port 21 = Z1 port
<23 : 22>	These bits are reserved.
<31 : 24>	When set to 1, each of these bits indicates that a channel parity error occurred on the corresponding port. 24 = X0 port 28 = Z0 port 25 = X1 port 29 = Z1 port 26 = Y0 port 30 = C-option (PE) port 27 = Y1 port 31 = I/O port
<43 : 32>	When set to 1, each of these bits indicates that a parity error occurred for the corresponding port RAM. 32 = X0 control RAM 38 = Y1 control RAM 33 = X0 data RAM 39 = Y1 data RAM 34 = X1 control RAM 40 = Z0 control RAM 35 = X1 data RAM 41 = Z0 data RAM 36 = Y0 control RAM 42 = Z1 control RAM 37 = Y0 data RAM 43 = Z1 data RAM
<45 : 44>	When set to 1, each of these bits indicates that a parity error occurred for the corresponding port RAM. 44 = PE port control RAM 45 = PE port data RAM
46	When set to 1, this bit indicates that a RAM parity error occurred on an incoming packet.

47	When set to 1, this bit indicates that a RAM parity error occurred on the routing tag look-up table.
48	When set to 1, this bit indicates that a RAM parity error occurred on the body of a response or request packet.
49	When set to 1, this bit indicates that a RAM parity error occurred on the head of a response packet.
50	When set to 1, this bit indicates that a RAM parity error occurred on a packet that transfers from the R option to the I option.
<55 : 51>	These bits are reserved.
56	When set to 1, this bit indicates that a packet misroute occurred.
57	This bit is not used.
58	When set to 1, this bit indicates that the network router detected an ECC error on the data that it is sending to the I option or the C option.
<63 : 57>	These bits are not used.

R_ERR1

The R_ERR1 register enables the error interrupts. When a bit of the R_ERR1 register is set to 1, the corresponding interrupt of the R_ERR0 register is enabled. When a bit of the R_ERR1 register is set to 0, the corresponding interrupt of the R_ERR0 register is disabled. Although an interrupt is disabled, the corresponding bit of the R_ERR0 register for that interrupt still indicates the state of the interrupt.

Figure 21. R Option Error Reporting



NOTE: The bold numbers correlate to the bits of the R_ERR0 register.

Offline Diagnostics

There are three offline diagnostic tests for the interconnect network: `luts`, `rchip`, and `xnet`.

luts

The routing tag look-up table test (`luts`) verifies the routing tag look-up tables and the functionality of their supporting logic.

rchip

The basic network router memory-mapped register test (`rchip`) verifies that the following network router memory-mapped registers function properly:

- Router software register (R_SW)
- Router channel configuration register (R_CHAN)

  NOTE: `rchip` does not test adaptive bit 22 and fast bit 23.
- Router virtual channel selection register (R_VCSEL)
- Router state register (R_STATE)
- Router SGET return path register (R_SGET_RTRN)
- Router port orient register (R_ORIENT)
- Router error register (R_ERR)
- Router reset control register (R_RESETS)

`rchip` is not an intensive test of the interconnect network.

xnet

The network test (`xnet`) verifies the functionality of the interconnect network by producing heavy traffic on the interconnect network. The heavy traffic causes conflicts among the packets that are traveling through the interconnect network.